9-14-2017

New Fractals for Computer Generated Art Created by Iteration of Polynomial Functions of a Complex Variable

Charles F. Babbs
Purdue University, babbs@purdue.edu

NEW FRACTALS FOR COMPUTER GENERATED ART CREATED BY ITERATION OF POLYNOMIAL FUNCTIONS OF A COMPELX VARIABLE

Charles F. Babbs, MD, PhD*

*Weldon School of Biomedical Engineering, Purdue University, West Lafayette, Indiana, USA

Abstract. Novel fractal forms can be created by iteration of higher order polynomials of the complex variable, z, with both positive and negative exponents, $z \leftarrow a_n z^{p_{max}} + a_{n-1} z^{p_{max}-1} + ... + a_0 z^0 + a_{-1} z^{-1} + a_{-2} z^{-2} + ... + a_{-p_{min}} z^{-p_{min}}$, followed by optional integer power transformation, $z \leftarrow z^k$. Such functions lead to an expanded universe of fascinating fractal patterns that can be incorporated into computer generated art.

Keywords. Algorithmic art, Chaos, Complex plane, Dynamics, Fractals, Generalized Mandelbrot set, Graphics, Iteration, Julia set, Mandelbrot set, Prisoner set.

Background

The fractal patterns in computer generated art are based on the concept of iteration. This concept can be demonstrated easily in just one dimension. Let f(x) be a function of real variable, x, and consider the sequence of computations x_0, $x_1 = f(x_0)$, $x_2 = f(x_1)$, $x_3 = f(x_2)$, and so on, such that the input or argument of f(x) in step n + 1 equals the output or value of f(x) in step n. This process in which $x_{n+1} = f(x_n)$ is called iteration or recursion.

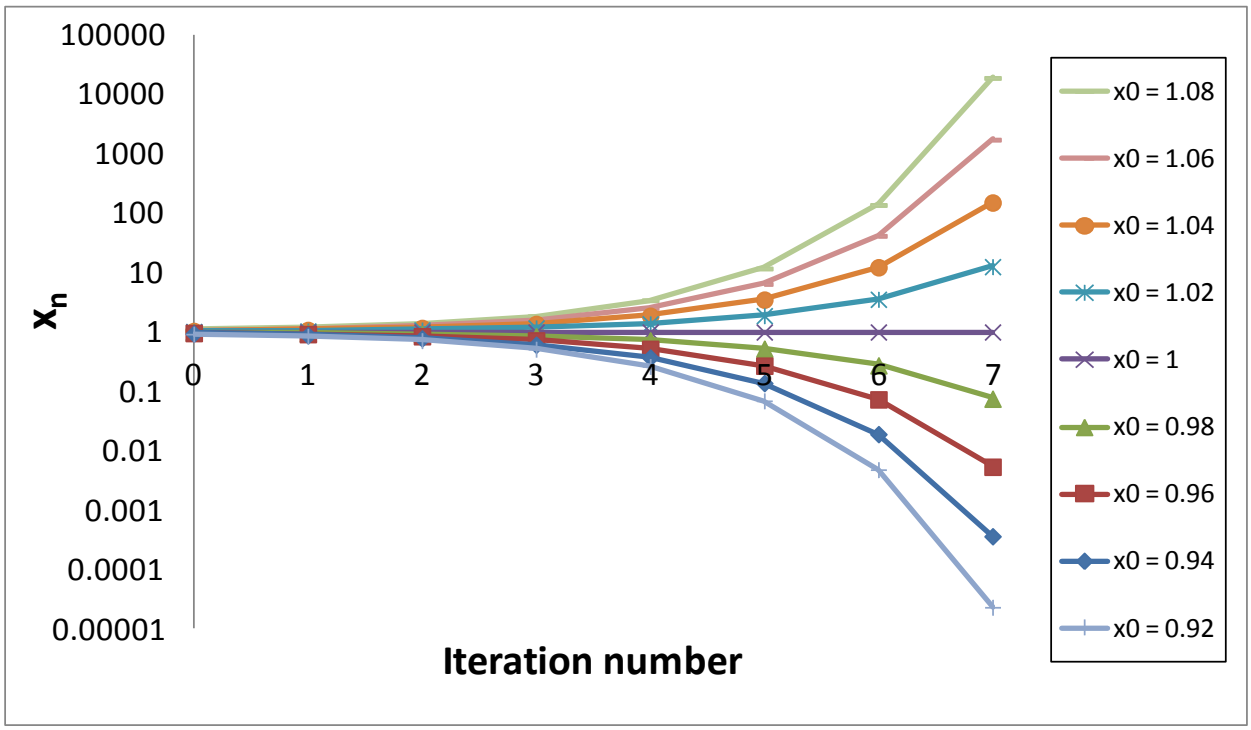


Figure 1. Example of iteration, a prisoner set, an escape set, and a Julia set in one dimension, x, which is represented on the vertical axis. Note log scale. Here $x_{n+1} = x_n^2$. Iterations with seed values $x_0 < 1$ rapidly dive toward zero. These seed values constitute the prisoner set. Iterations with seed values $x_0 > 1$ rapidly climb toward infinity. These seed values constitute the escape set. The boundary at point $x_0 = 1$ is the Julia set.

Consider the following one-dimensional example. Suppose $f(x) = x^2$, and $x_0 = 2$. Then, as $n \to \infty$, $x_n \to \infty$. However, if $x_0 = ½$, then as $n \to \infty$, $x_n \to 0$. The set of all seed values, x_0, for which for which the iterated values $x_n \to \infty$ is called the escape set. The set of all seed values of x_0 for which iterated values, x_n, remain finite is called the prisoner set. As shown in Figure 1 for this one-dimensional example, the escape set includes all $x_0 > 1$, and the prisoner set includes all $x_0 < 1$. The boundary between the escape set and the prisoner set is called the Julia set. For the one-dimensional example in Figure 1 the Julia set is{1}. In general the prisoner set is the set of seed values, x_0, for which the iterated values, x_n, remain bounded as n approaches infinity. The escape set is the set of seed values, x_0, for which the iterated values, x_n, are unbounded as n approaches infinity. The Julia set includes any points at the border between the prisoner set and the escape set.

When this process is extended to two dimensions the Julia sets in some cases become extremely intricate and infinitely long curves in the plane showing increasingly finer, but self-similar detail, no matter how much a region of the plane is magnified. Such an infinitely long curve is called a fractal. Many beautiful fractal-like patterns can be found in nature, including trees, mountains, and coastlines.

Classically, fractals are generated mathematically in two dimensions by the iteration of functions of complex numbers, $z = x + iy$, for ordinary, real numbers x and y and $i = \sqrt{-1}$. Complex numbers can be represented in the complex plane with x-values on the horizontal axis and y-values on the vertical axis. That is, the complex number, z, can be represented by point (x, y) in the complex plane. The "real part", x, of the complex number is plotted on the horizontal, or "real", axis of the complex plane, and the "imaginary part", y, of the complex number, the part that is multiplied by $i = \sqrt{-1}$, is plotted on the vertical, or "imaginary", axis. Iteration is done by plotting successive points $z_{n+1} = f(z_n)$ in the complex plane.

For example, a Mandelbrot-like set can be created by iteration of

$$z_{n+1} = z_n^2 + c \text{, or in alternative notation, } z \leftarrow z^2 + c \qquad (1)$$

for complex constant, $c = x_c + iy_c$. Since

$$z_n^2 = x_n^2 + 2x_n y_n i - y_n^2 = \left(x_n^2 - y_n^2\right) + \left(2x_n y_n\right) i \text{ ,} \qquad (2)$$

the iteration of the complex number in this way always produces another complex number and is equivalent to the two dimensional iteration

$$x_{n+1} = x_n^2 - y_n^2 + x_c \qquad (3a)$$

$$y_{n+1} = 2x_n y_n + y_c \text{ .} \qquad (3b)$$

Using this rule, it is easy to compute a two dimensional image of the prisoner set along the following lines. Define a grid-like two-dimensional array of discrete points, x_0, y_0, in a bounded region of the complex plane, usually near the origin, with each point representing a picture element (pixel) in the final image. For each particular x_0, y_0, begin the iteration process $z_{n+1} = f(z_n) = f_1(x_n, y_n) + f_2(x_n, y_n)\, i$ for iterations n = 0, 1, 2, 3, ..., n_{max}. Continue iteration until either the magnitude of z diverges beyond a certain preset limit, R, or until the number of iterations equals a maximum allowable number of iterations, n_{max}, such as 20. Specifically, if $x_n^2 + y_n^2 > R^2$ for the real boundary, R, indicating the "practical escape radius", then stop iteration and assign point (x_0, y_0) to the escape set. Otherwise, if $n = n_{max}$ and $x_n^2 + y_n^2 < R^2$, then assign point (x_0, y_0) to the prisoner set. Furthermore, for points x_0, y_0 in the escape set use the last tested value of n (call this value n*) as a measure of the escape speed. Smaller values of n* indicate faster escape.

Using n* in this way, the complex plane can be color coded by escape speed for artistic effect. If $n^* = n_{max}$, then the pixel centered at x_0, y_0 is colored to represent the prisoner set. If $n^* < n_{max}$, then the pixel centered at x_0, y_0 is colored differently as a function of n*. In this way the colors of points outside the prisoner set are assigned according to escape speed in a way that highlights the boundary of the prisoner set.

For any iterated function of the complex variable, f(z) = f(x + yi), a computer program can test all possible seed values, x_0, y_0 in a selected region of the complex plane to check for escape behavior, and the corresponding points can be color coded by escape speed. A general algorithm for making such escape speed images is listed in Box 1.

Box 1. General computational algorithm for fractal image generation

set escape radius, R, and maximum number of iterations, n_{max}

read z-plane dimensions, z-plane resolution, complex polynomial coefficient values, and color parameters

open image file, write image name and format specifications

for each pixel

    determine coordinates x_0, y_0 in the complex plane

    for n = 0 to $n_{max} - 1$

        compute x_{n+1} and y_{n+1} using specified functions

        if $x_{n+1}^2 + y_{n+1}^2 > R^2$, then exit this for loop

    next n

    assign desired red, green, and blue (RGB) color levels (0 to 255) to selected pixel as specified functions, $f_R(n)$, $f_G(n)$, $f_B(n)$

    write RGB coded image data in a convenient format

next pixel

close image file

Figure 2 shows a set of escape speed images for iteration of the complex number $z \leftarrow z^2 + c$, with tunable complex parameter, $c = x_c + iy_c$. The images are reminiscent of the famous Mandelbrot set. However, note that the images of Figure 2 are not identical to those of the classical Mandelbrot set because the values at each x, y location in the image plane are assigned according to the variables x_0 and y_0, rather than to variables x_c and y_c (with $x_0 = y_0 = 0$), as is done in the most common iteration scheme for generating the Mandelbrot set. That is to say the fractals in Figure 2 are plotted in the x_0, y_0 plane not in the x_c, y_c plane, or using the terminology of Gujar and Bhavsar (1991), they are z-plane fractals, rather than c-plane fractals. For z-plane fractals, changes in the constant, c, lead to variety of alternative Mandelbrot-like sets, as shown in Figures 2(a) through 2(d). This same style of iteration was described by previously by Norton (1989) and is used exclusively in the present paper.

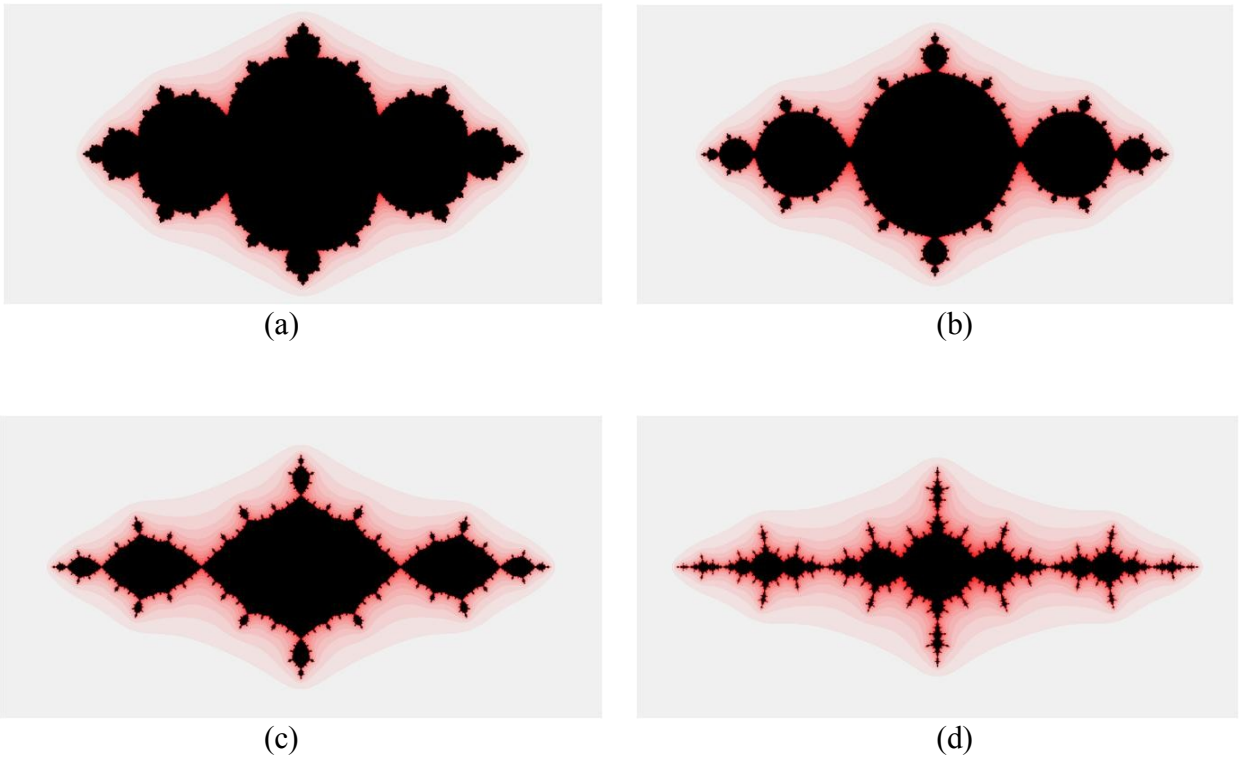


Figure 2. Escape-time fractal in the complex plane for iteration of $z \leftarrow z^2 + c$. Practical escape radius, R = 8.4. The number of the last completed iteration, n*, is the iteration number when escape distance from the origin becomes > R, or if there is no escape in n_{max} iterations, then $n^* = n_{max}$. Prisoner set is black ($n^* = 20$); near field escape set is pink $5 \le n^* < 20$; far field escape set is gray ($n^* < 5$). The complex constant $c = x_c + iy_c$ is a tunable parameter. In (a) $x_c = -0.7$, $y_c = 0$. In (b) $x_c = -0.9$, $y_c = 0$. In (c) $x_c = -1.1$, $y_c = 0$. In (d) $x_c = -1.3$, $y_c = 0$. Numerically, the horizontal range of the images extends from x = –2 to x = 2. The vertical range extends from y = –1 to y = 1.

The patterns in Figure 2 are true fractals, as is shown, for example, for the form in Figure 2(b) by the expanded scale and higher resolution representation shown in Figure 3.

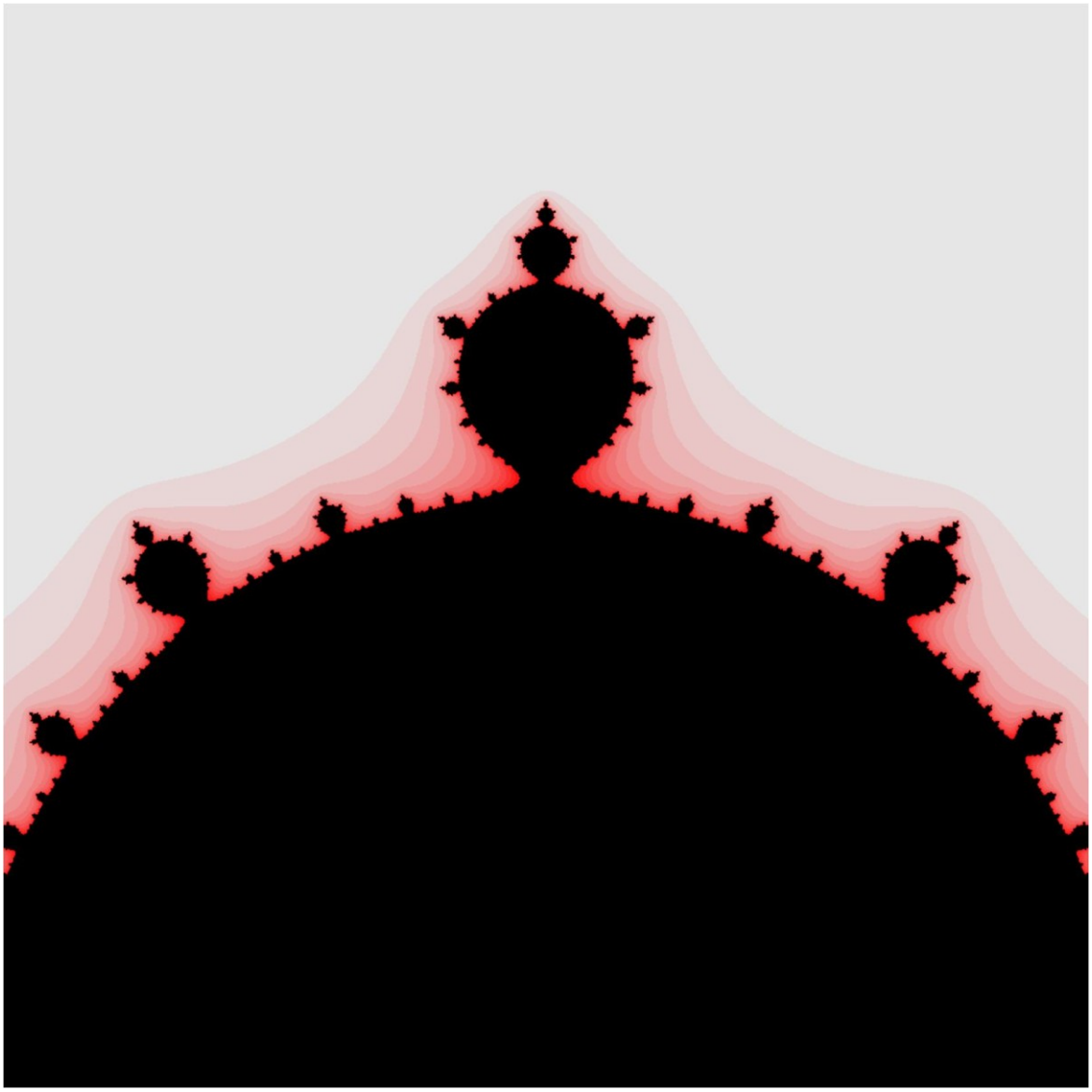


Figure 3. Escape-time fractal in the complex plane for iteration of $z \leftarrow z^2 - 0.9$. Prisoner set is black; near field escape set is pink; far field escape set is gray. Numerically, the horizontal range of the image extends from $x = -0.5$ to $x = 0.5$. The vertical range extends from $y = 0$ to $y = 1$.

Figures 4 and 5 show even higher resolution and more expanded views of the fractal pattern in Figure 3. Figure 4 represents 20-fold magnification of Figure 2(b). Figure 5 represents 200-fold magnification. The fractal pattern is preserved after magnification by more than two orders of magnitude.

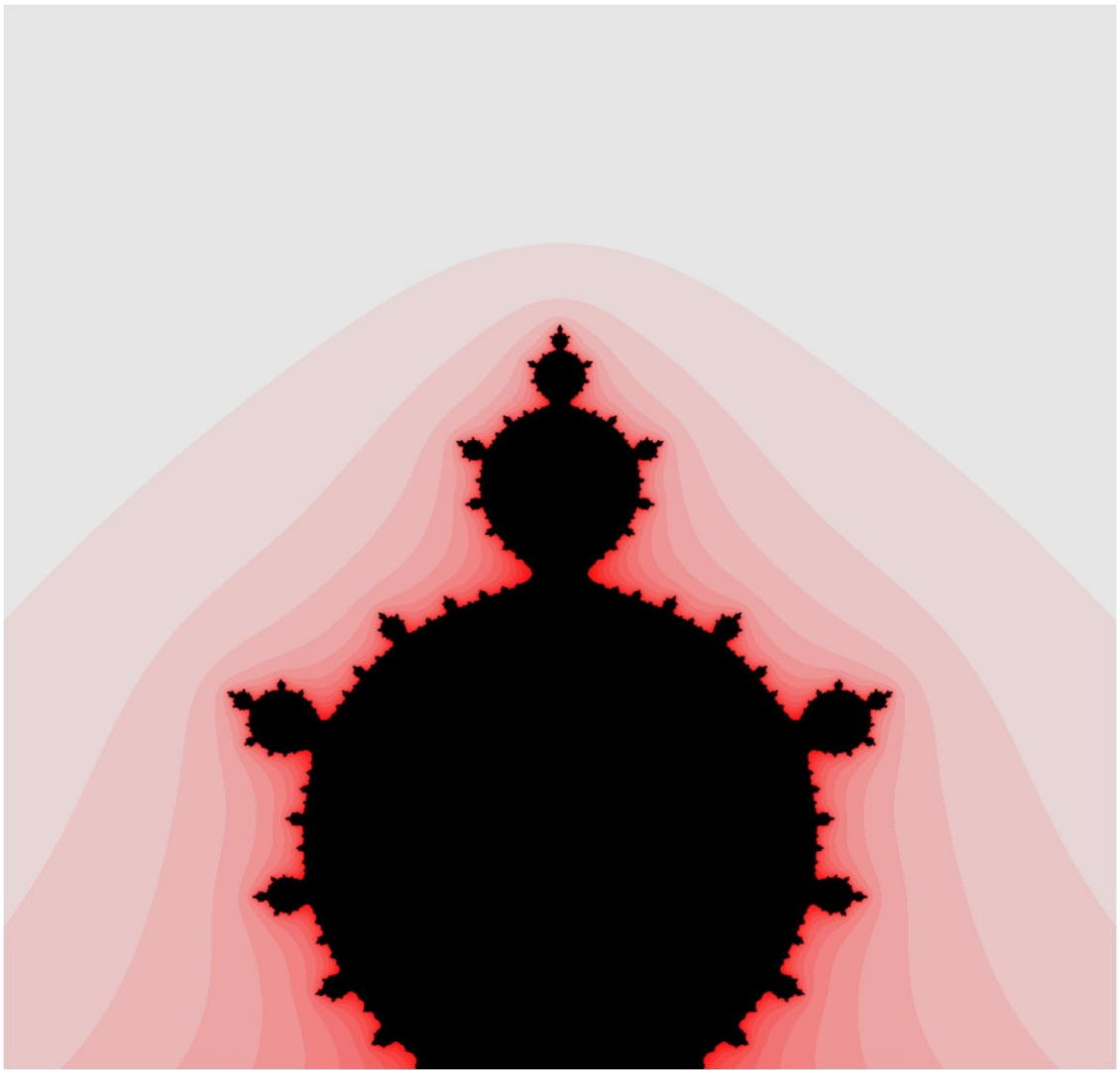


Figure 4. Escape-time fractal in the complex plane for iteration of $z \leftarrow z^2 - 0.9$. Prisoner set is black; near field escape set is pink; far field escape set is gray. Numerically, the horizontal range of the image extends from $x = -0.05$ to $x = 0.05$. The vertical range extends from $y = 0.75$ to $y = 0.85$.

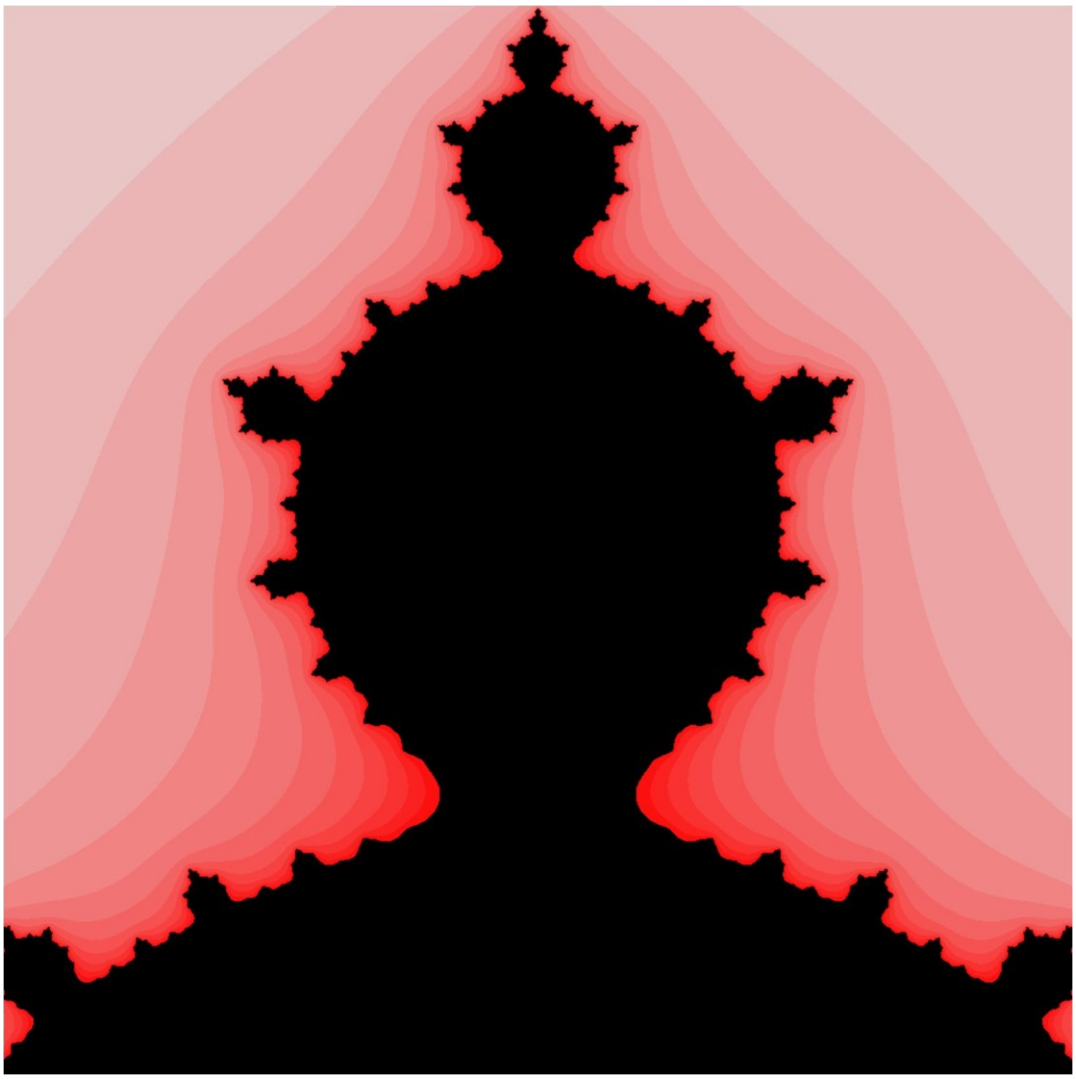


Figure 4. Escape-time fractal in the complex plane for iteration of $z \leftarrow z^2 - 0.9$. Prisoner set is black; near field escape set is pink; far field escape set is gray. Numerically, the horizontal range of the image extends from x = –0.005 to x = 0.005. The vertical range extends from y = 0.81 to y = 0.82.

The present faculty working paper shows students how to create a rich variety of complex and interesting fractal forms using higher order polynomial functions beyond the realm of $z_{n+1} = z_n^2 + c$.

Fractals created by iteration of polynomial functions of the complex variable, z

Consider iteration of a generalized polynomial function of the complex variable, z,

$$z \leftarrow a_n z^{p_{max}} + a_{n-1} z^{p_{max}-1} + ... + a_0 z^0 + a_{-1} z^{-1} + a_{-2} z^{-2} + ... + a_{-p_{min}} z^{-p_{min}}, \quad (4)$$

including both positive and integer negative powers of z and complex constants, a_p, for $p_{max} \geq p \geq -p_{min}$. Here we show that a rich variety of fractal patterns can be created by iteration of such higher order polynomial functions of a single complex variable using a simple computer program to determine prisoner sets, escape sets, and color coded escape speeds. Although not all polynomial functions will generate fractal patterns; it is interesting to explore the properties of functions that do.

The complex arithmetic needed to specify successive iterations of Equation (4) in computer code requires functions that implement complex addition, multiplication, and inversion (Appendix). For addition and subtraction we have

$$(a + bi) \pm (c + di) = (a \pm c) + (b \pm d)i. \quad (5)$$

For multiplication we have

$$(a + bi) \cdot (c + di) = ac + bci + adi + bdi^2 = (ac - bd) + (bc + ad)i. \quad (6)$$

For inversion we multiply by unity as follows,

$$\frac{1}{a + bi} = \frac{1}{a + bi} \cdot \frac{a - bi}{a - bi} = \frac{a - bi}{a^2 + b^2} = \left(\frac{a}{a^2 + b^2}\right) - \left(\frac{b}{a^2 + b^2}\right)i. \quad (7)$$

By implementing the above functions for complex arithmetic and utilizing color coded escape speed, as described above, a rich variety of fractal patterns can be obtained from the iteration of general polynomial functions of a single complex variable, z. The following is a small sample.

Results

Polynomial with positive powers of z

Figure 6 shows the color coded prisoner set (black) and near field and far field escape sets (pink and gray) for iteration of $z \leftarrow a_0 + a_7 z^7$, where complex constant $a_0 = 0.6 + 0.6i$ and $a_7 = 1$. The 7-way branching pattern is apparent. The magnified view in Figure 7 confirms the fractal nature of the pattern. The local structure of the prisoner set is dominated by families of branched, 7-fold radial outcroppings that become smaller in successive generations, *ad infinitum*.

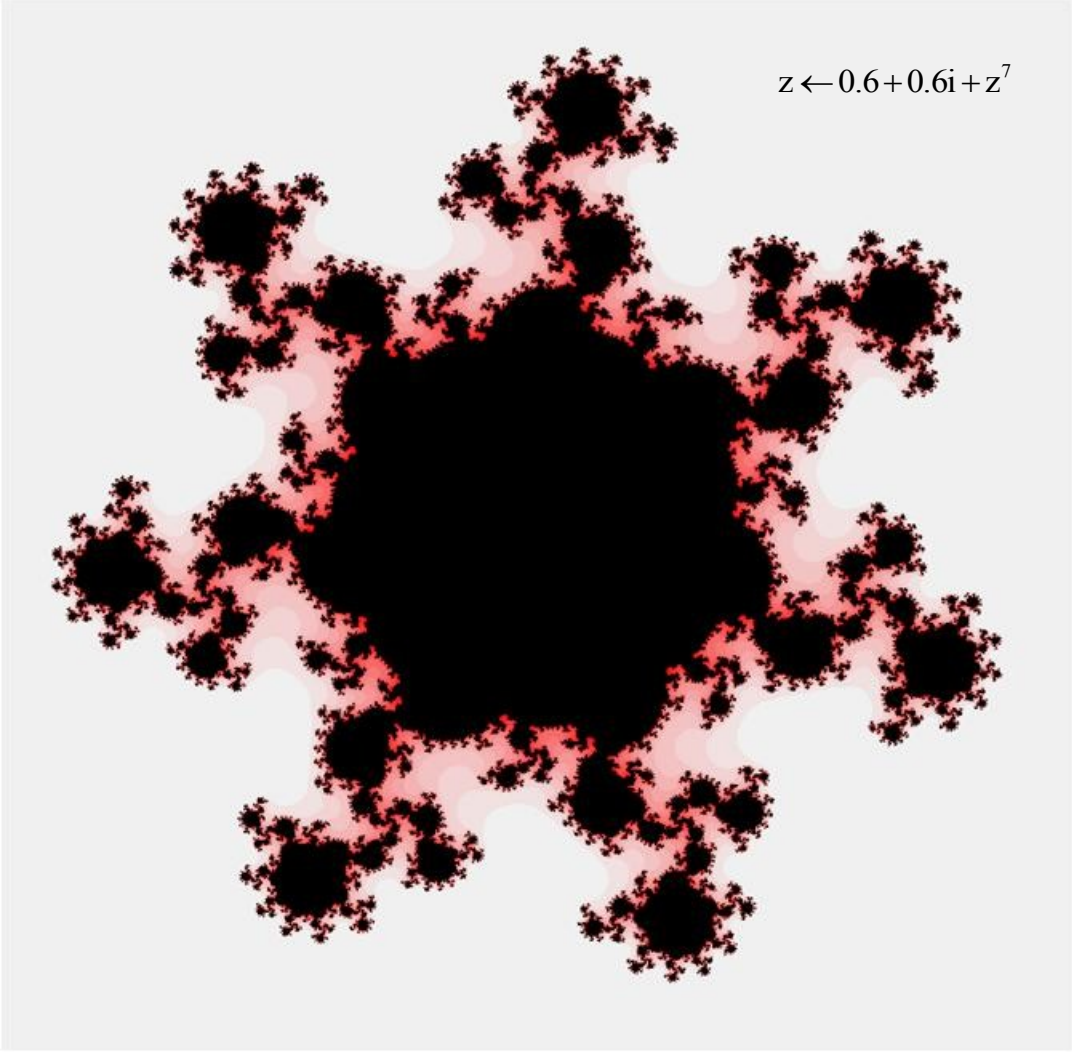


Figure 6. Color coded prisoner set (black) and near field and far field escape sets (pink and gray) for iteration of $z \leftarrow a_0 + a_7 z^7$, where complex constants $a_0 = 0.6 + 0.6i$ and $a_7 = 1$. Numerically the horizontal span of the image ranges from –1.2 to 1.2 on the real axis. The vertical span of the image ranges from –1.2 to 1.2 on the imaginary axis.

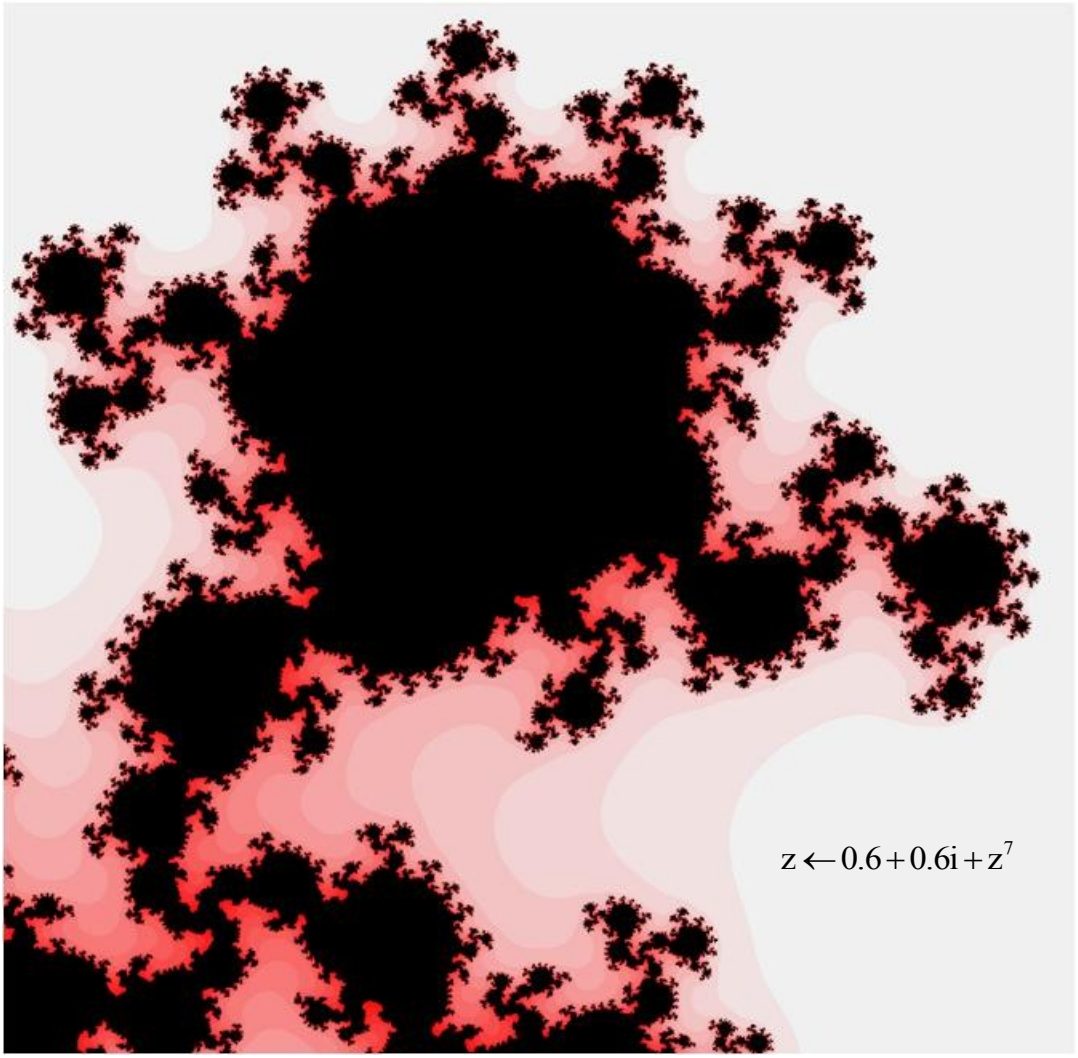


Figure 7. Magnified region of the top most lobe of the fractal in Figure 6. Numerically the horizontal span of the image ranges from –0.07 to 0.33 on the real axis. The vertical span of the image ranges from 0.7 to 1.1 on the imaginary axis.

Polynomials with negative powers of z

Figure 8 is a representative fractal pattern generated from a polynomial with negative powers of z, namely $z \leftarrow 0.4 + \left(1 + z^{-1} + z^{-2} + z^{-3}\right)i$. The prisoner set (black) looks connected. However the escape set (gray to white) consists of infinitely many islands, each with a globular structure. Figure 9 shows the top cluster in Figure 8 at 4X magnification.

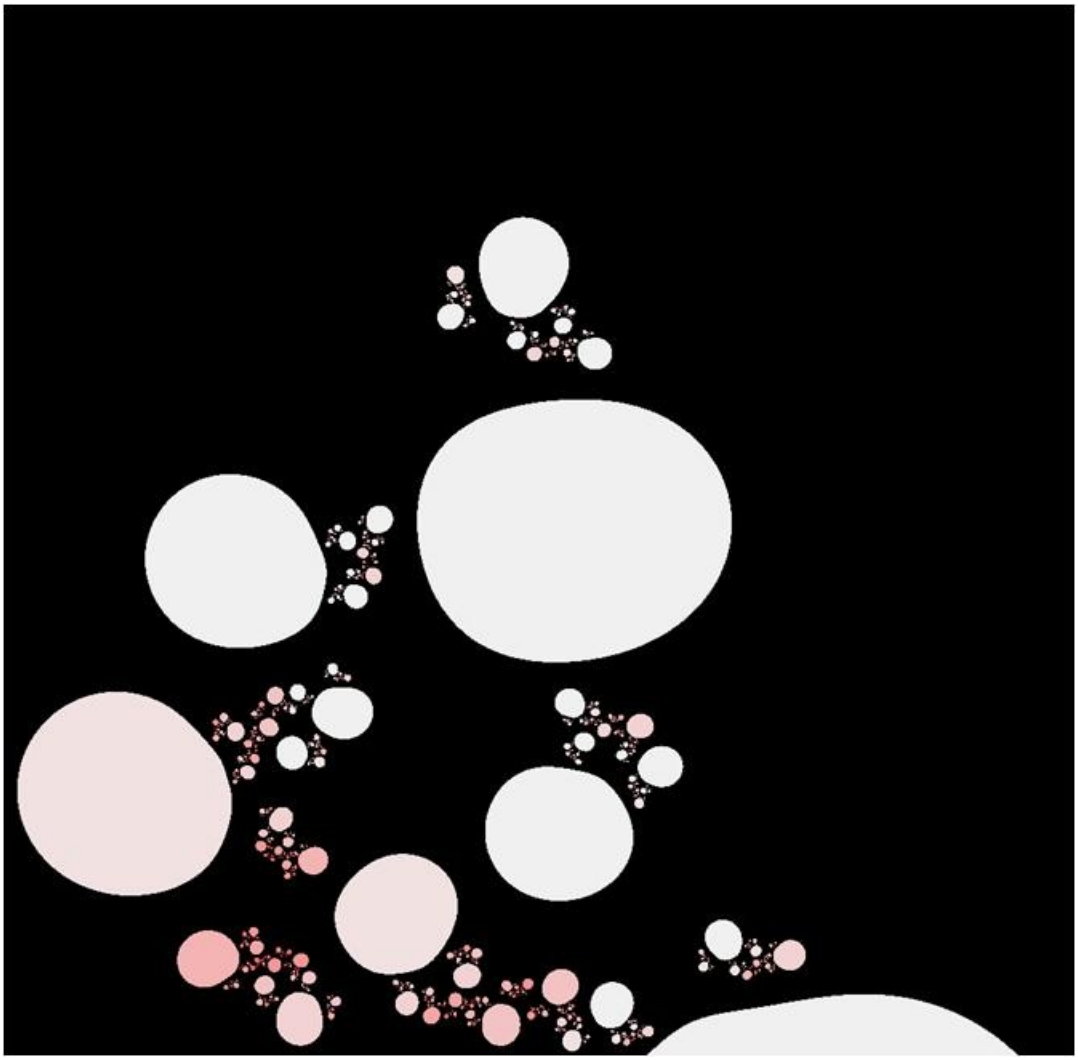


Figure 8. Fractal pattern for iteration of $z \leftarrow a_0 + a_{-1}z^{-1} + a_{-2}z^{-2} + a_{-3}z^{-3}$ where complex constants $a_0 = 0.4 + i$ and $a_{-1} = a_{-2} = a_{-3} = i$ or $z \leftarrow 0.4 + \left(1 + z^{-1} + z^{-2} + z^{-3}\right)i$. Numerically the horizontal span of the image ranges from –2 to 2 on the real axis. The vertical span of the image ranges from –2 to 2 on the imaginary axis. Prisoner set is black. Far field escape set is gray. Near field escape set is pink.

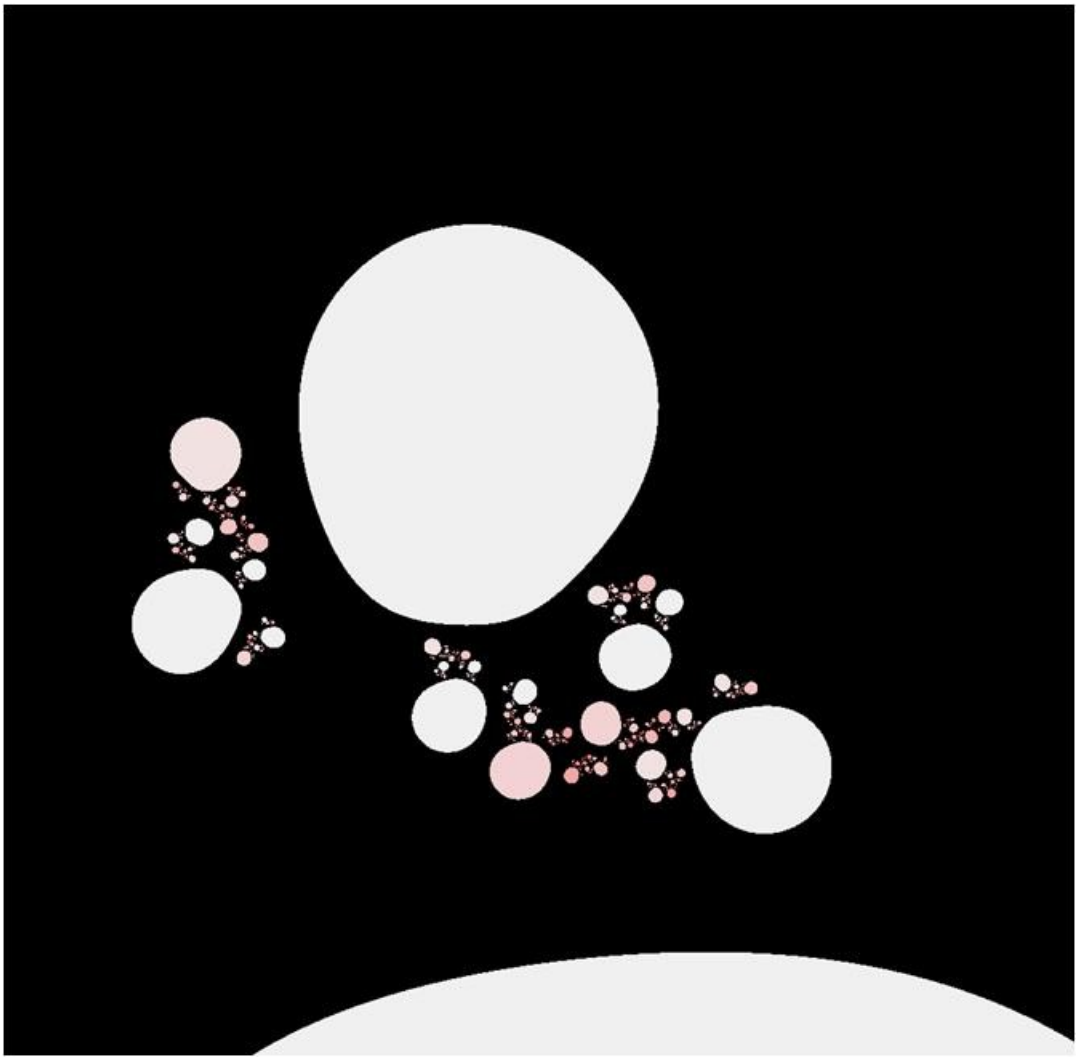


Figure 9. Top part of fractal pattern in Figure 8 for iteration of $z \leftarrow a_0 + a_{-1}z^{-1} + a_{-2}z^{-2} + a_{-3}z^{-3}$, where complex constants $a_0 = 0.4 + i$ and $a_{-1} = a_{-2} = a_{-3} = i$ or $z \leftarrow 0.4 + \left(1 + z^{-1} + z^{-2} + z^{-3}\right)i$. Numerically the horizontal span of the image ranges from –0.5 to 0.5 on the real axis. The vertical span of the image ranges from 0.4 to 1.4 on the imaginary axis. Prisoner set is black. Far field escape set is light gray. Near field escape set is pink.

Shirriff (1993) has suggested that when negative powers are used in the iterated function to generate fractals, much of the structure of the fractals may be an artifact of the stopping criterion. This suggestion deserves further investigation. For example, increasing the escape radius, R, or measuring escape speed in terms of total orbit length for a fixed number of iterations may have a stabilizing effect. Figure 10 shows the same expanded area of the complex plane as that in Figure 9. The only difference is that the practical escape radius has been increased from R = 7 to R = 20 units from the origin, making escape more difficult to achieve in a fixed maximum number of iterations, in this case 20. With this more strict escape criterion the effective prisoner set expands somewhat, but the essential fractal pattern remains.

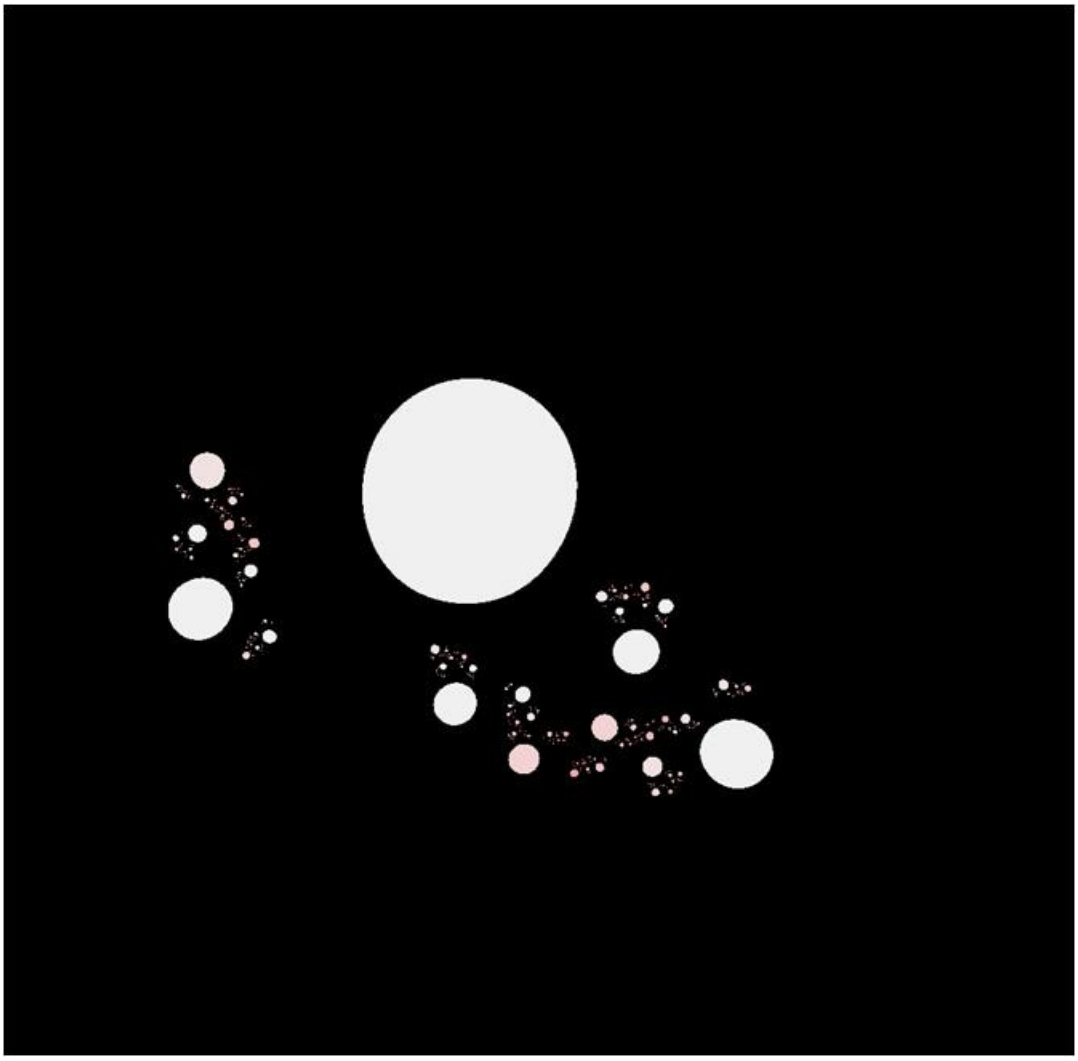


Figure 10. Re-run of the iteration in Figure 9 with more stringent criterion for escape. Here, compared to Figure 9, the practical escape radius has been increased from 7 to 20 units from the origin, making effective escape more difficult. The apparent prisoner set (black) is expanded.

Polynomials with both positive and negative powers of z

The present computational scheme, based on Equation (4) allows for iteration of functions containing both positive and negative integer powers of a complex variable. Figure 11 shows the result of iteration of the expression $z_{n+1} = z_n^2 + \frac{1}{2z_n^2} - \frac{3}{4}$, which reveals a Mandelbrot-like pattern appearing in multiple islands, *ad infinitum*, in effect multiplying the original infinities of $z \leftarrow z^2 + c$, as shown with magnification in Figure 12. Here the prisoner set consists of infinitely many islands, each with an infinite coastline resembling a classical Mandelbrot set.

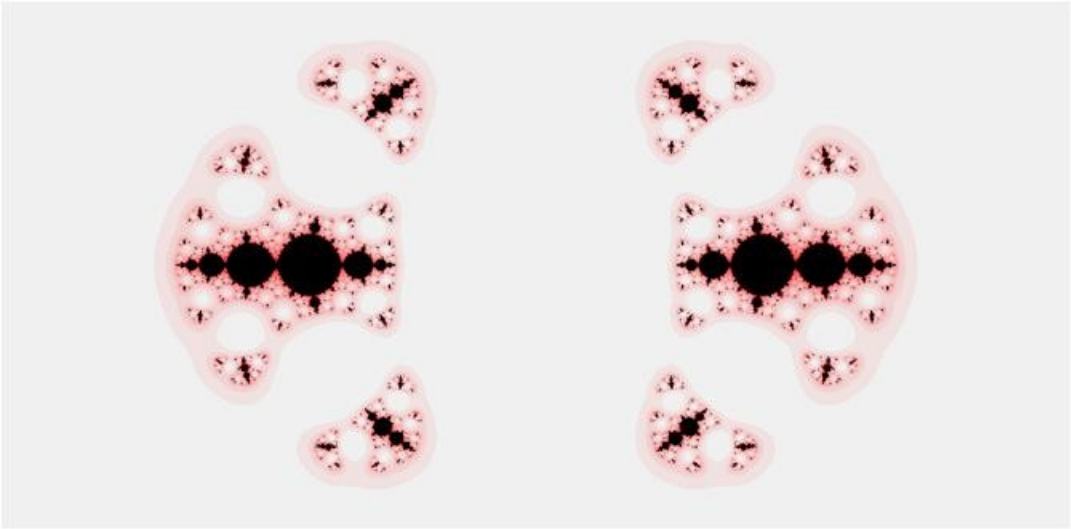


Figure 11. Iteration of $z \leftarrow a_0 + a_2 z^2 + a_{-2} z^{-2}$ where complex constants $a_0 = -0.75$, $a_2 = 1$, and $a_{-2} = 0.5$ or $z \leftarrow z^2 + 0.5z^{-2} - 0.75$. Numerically the horizontal span of the image ranges from −2 to 2 on the real axis. The vertical span of the image ranges from −1 to 1 on the imaginary axis. Prisoner set is black. Far field escape set is gray. Near field escape set is pink.

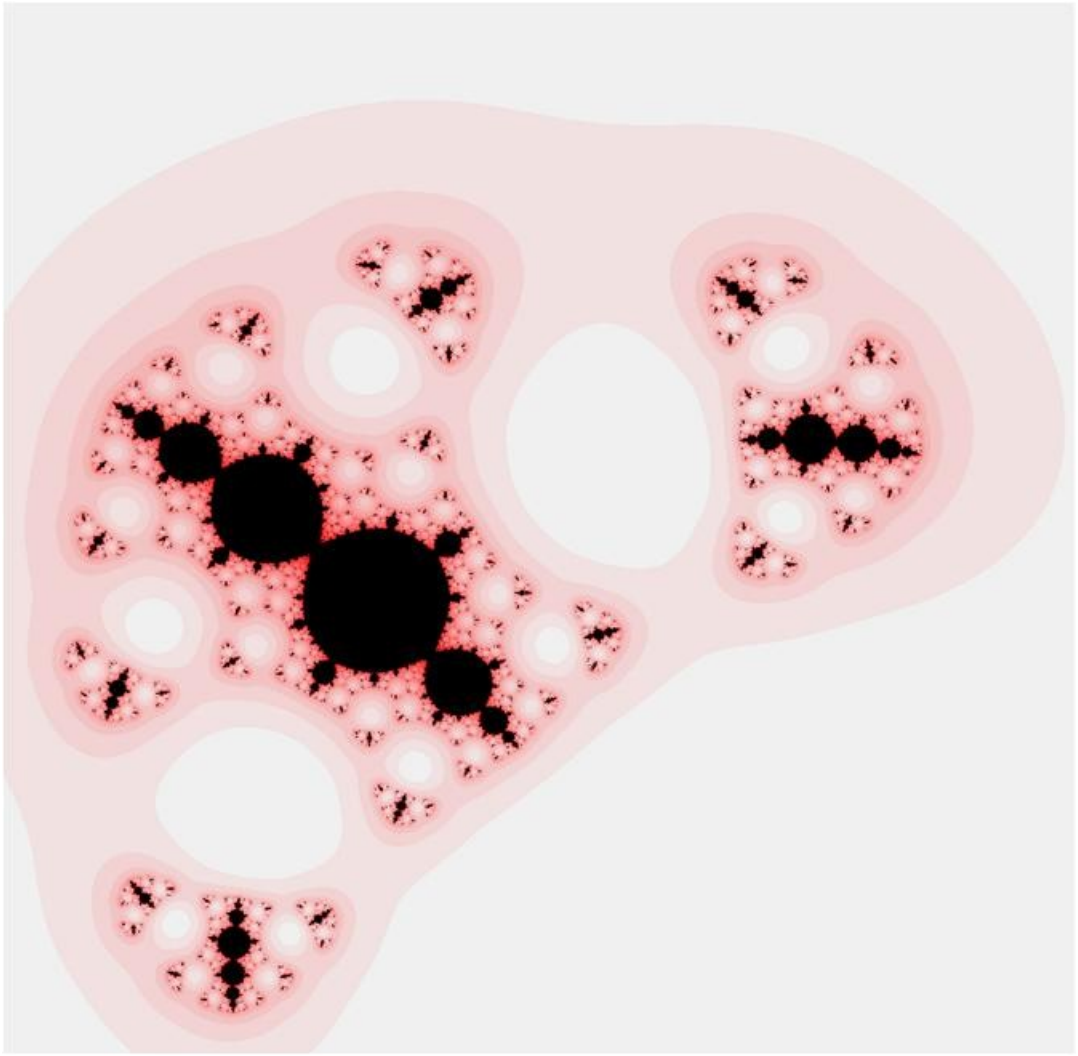


Figure 12. Magnified region of Figure 11 from iteration of $z \leftarrow a_0 + a_2 z^2 + a_{-2} z^{-2}$ where complex constants $a_0 = -0.75$, $a_2 = 1$, and $a_{-2} = 0.5$ or $z \leftarrow z^2 + 0.5z^{-2} - 0.75$. Numerically the horizontal span of the image ranges from 0.4 to 0.9 on the real axis. The vertical span of the image ranges from 0.4 to 0.9 on the imaginary axis. Prisoner set is black. Far field escape set is gray. Near field escape set is pink.

Figure 13 shows another example of this multiplying effect of summed positive and negative powers with seven way branched symmetry. The iteration rule is $z \leftarrow z^7 / 400 + z^{-7} + 1$. The fractals generated have both rotational and reflection symmetries.

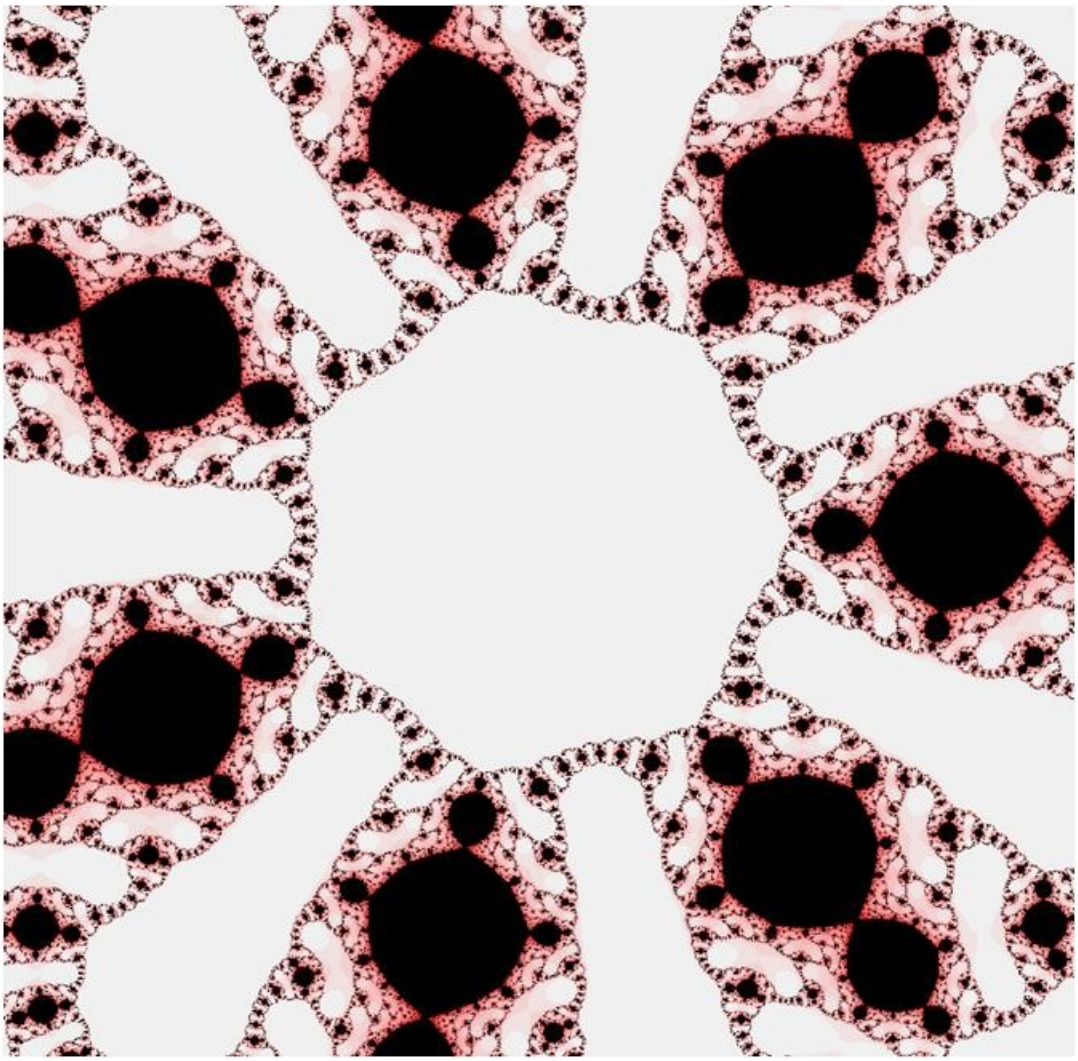


Figure 13. Iteration of $z \leftarrow a_0 + a_7 z^7 + a_{-7} z^{-7}$, where complex constants $a_0 = 1$ and $a_7 = 0.0025$, and $a_{-7} = 1$ or $z \leftarrow 0.0025 z^7 + z^{-7} + 1$. Numerically the horizontal span of the image ranges from –2 to 2 on the real axis. The vertical span of the image ranges from –2 to 2 on the imaginary axis. Prisoner set is black. Far field escape set is gray. Near field escape set is pink. R = 7, $n_{max} = 20$.

Raising the complete polynomial to an integer power

Figure 14 shows the result of squaring of the function in Figure 13, namely $z \leftarrow \left(0.0025z^7 + z^{-7} + 1\right)^2$. This is a specific example of a power transformation $z_{n+1} \leftarrow z_{n+1}^k$ with positive or negative integer power, k.

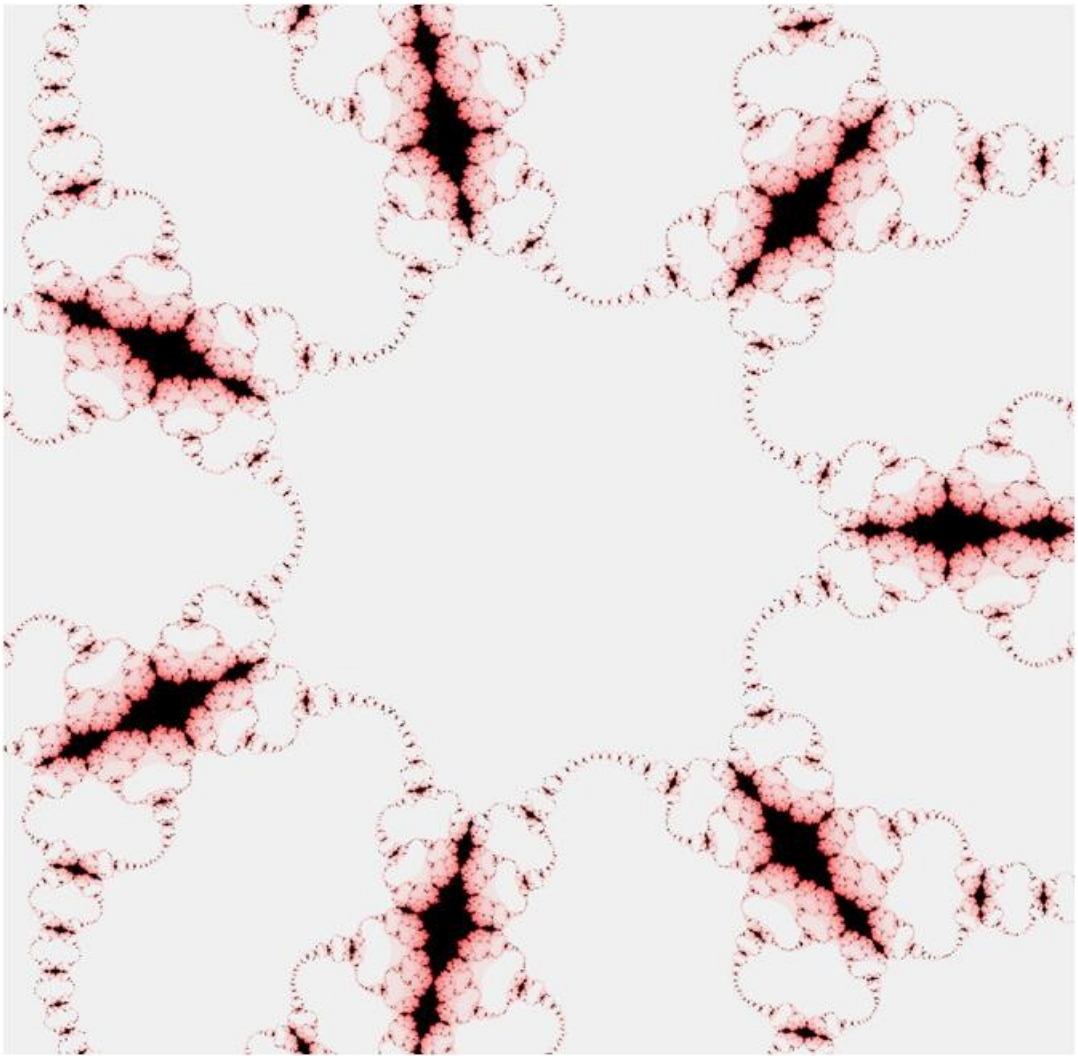


Figure 14. Iteration of $z \leftarrow \left(a_0 + a_7 z^7 + a_{-7} z^{-7}\right)^2$. Details similar to Figure 13.

Figures 15 and 16 show a more striking result, when $z_{n+1} = 6 + 0.0025z_n^7 + z_n^{-7}$ is inverted or raised to the −1 power. A completely new pattern is revealed. And as shown in Figure 16, it is indeed fractal in nature. Multiple copies of 7-tuples of self-similar islands begin to appear with magnification. The central pattern or "nucleus of the cell" in Figure 16 is a self-similar representation of the original, large scale form. The banding pattern of the ring structure in Figure 15 is also self-similar at multiple scales, as can be seen by exploring the region $-0.05 \le x \le 0.05$, $-2 \le y \le -1.9$, and in turn, $-0.005 \le x \le 0.005$, $-1.68 \le y \le -1.67$ (images not shown).

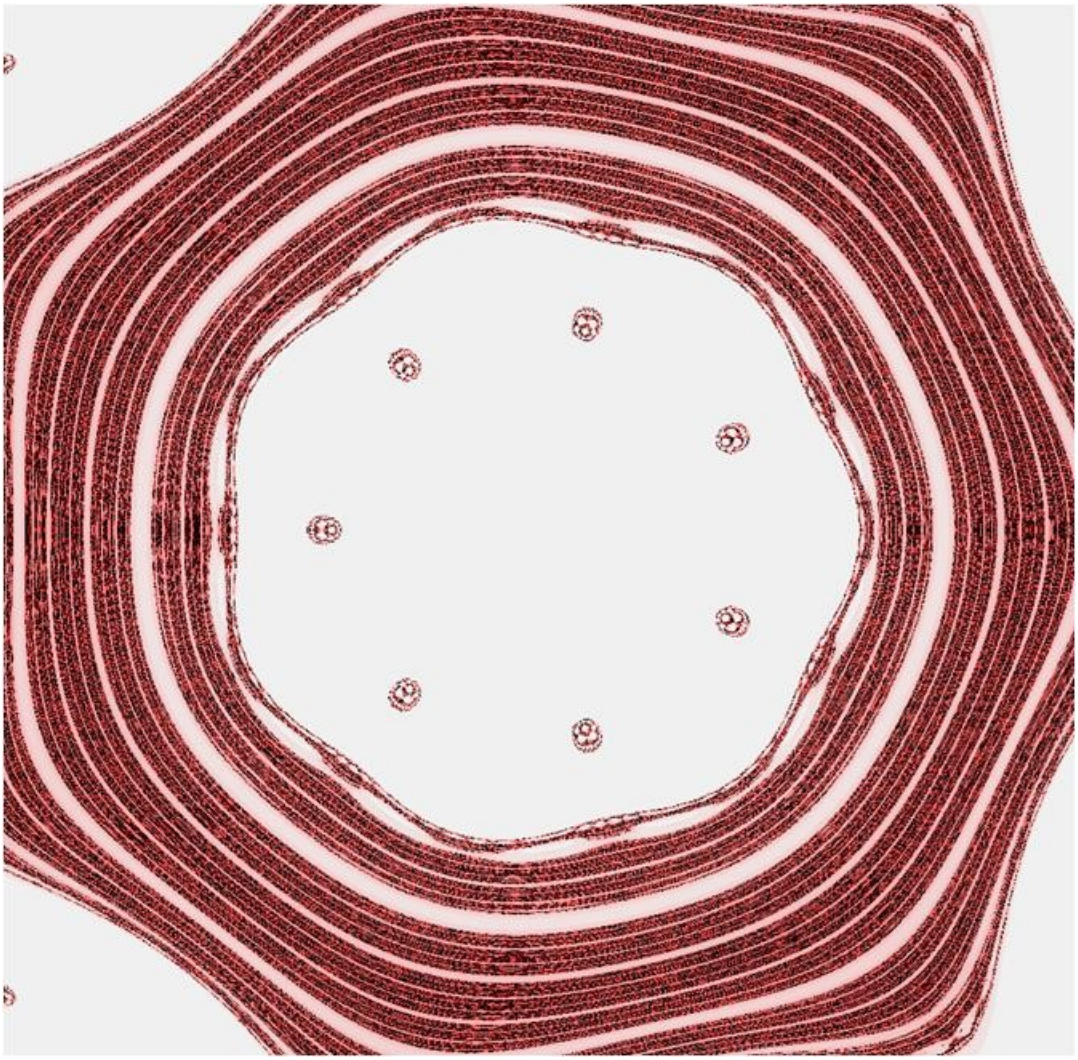


Figure 15. Iteration of $z \leftarrow \left(6 + 0.0025z^7 + z^{-7}\right)^{-1}$. Numerically the horizontal span of the image ranges from −2 to 2 on the real axis. The vertical span of the image ranges from −2 to 2 on the imaginary axis. Prisoner set is black. Far field escape set is gray. Near field escape set is pink. R = 7, n_{max} = 20.

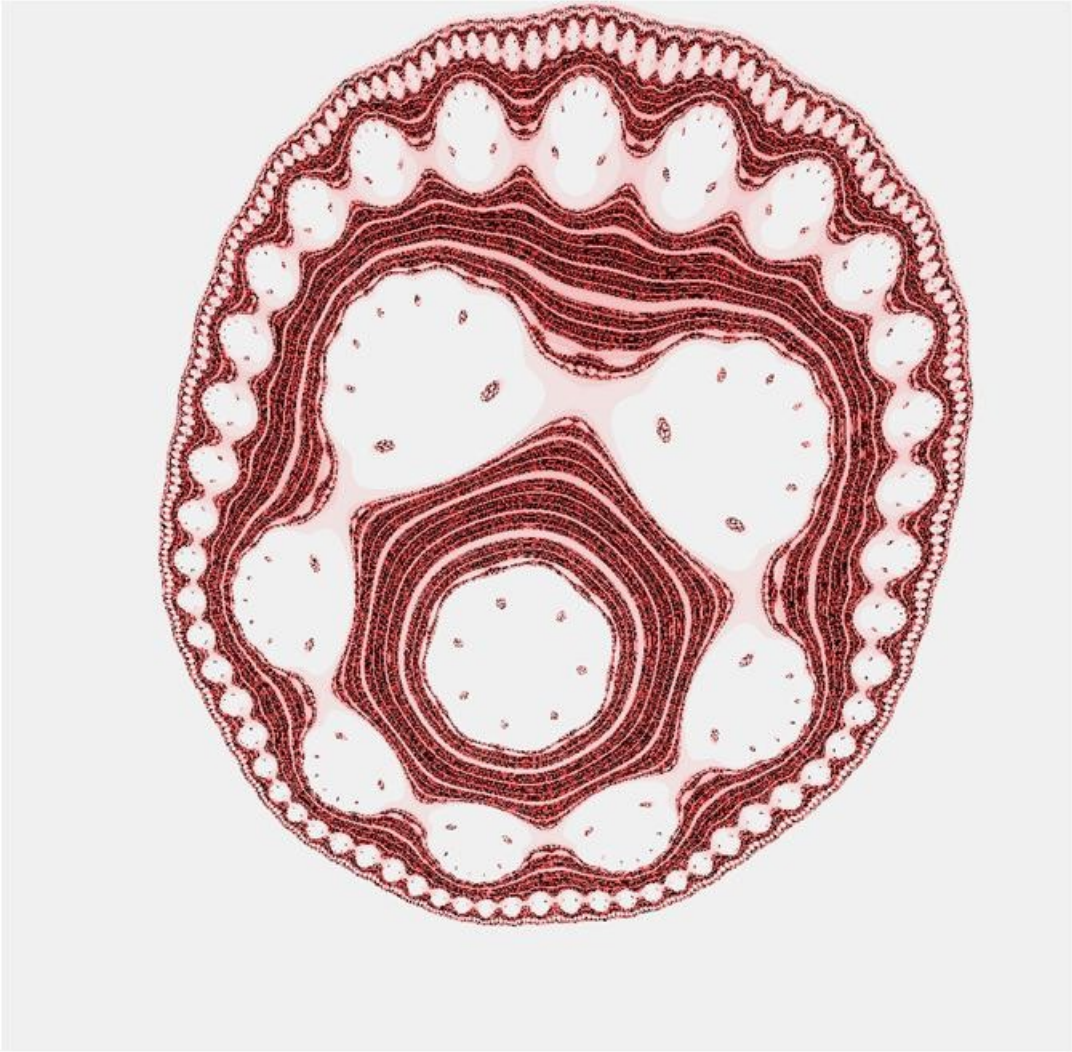
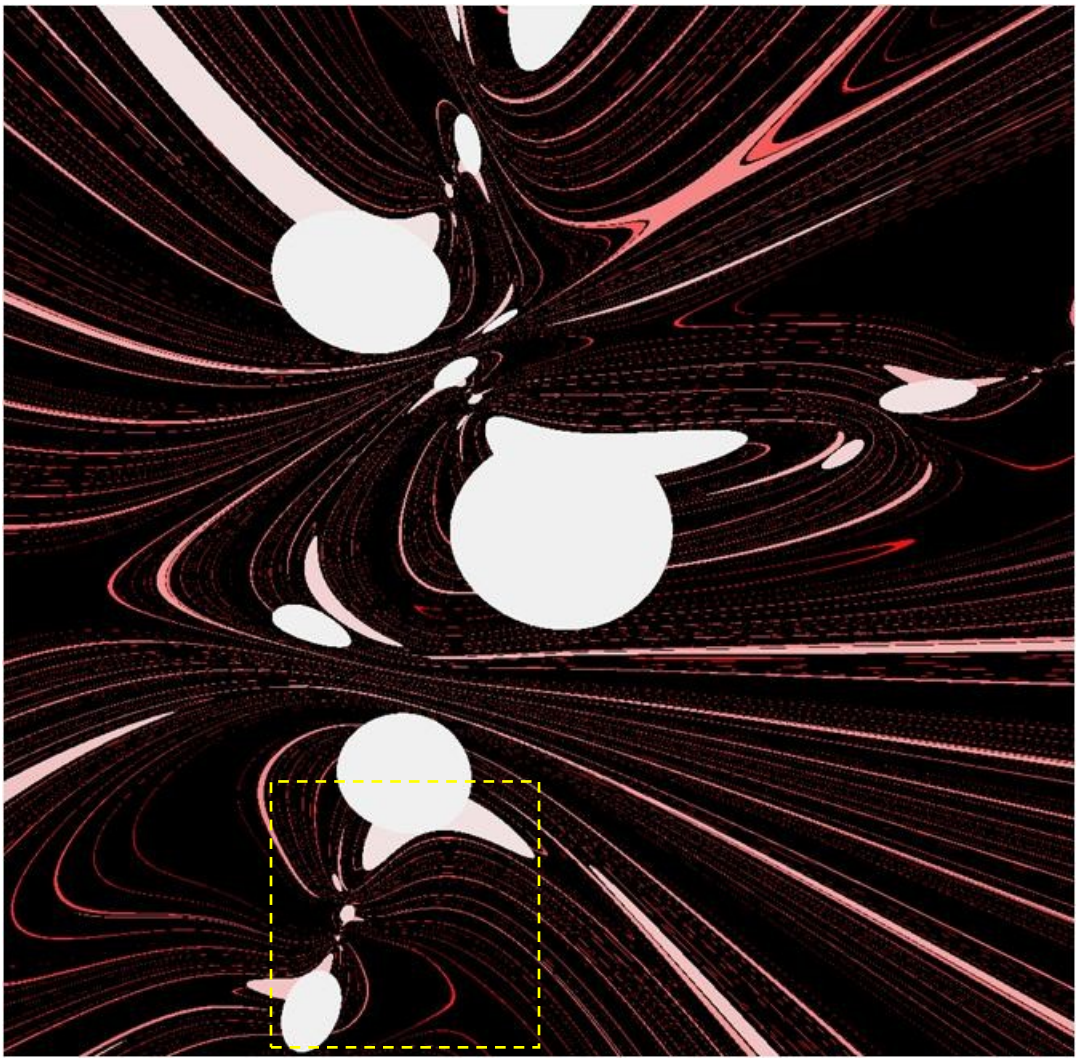
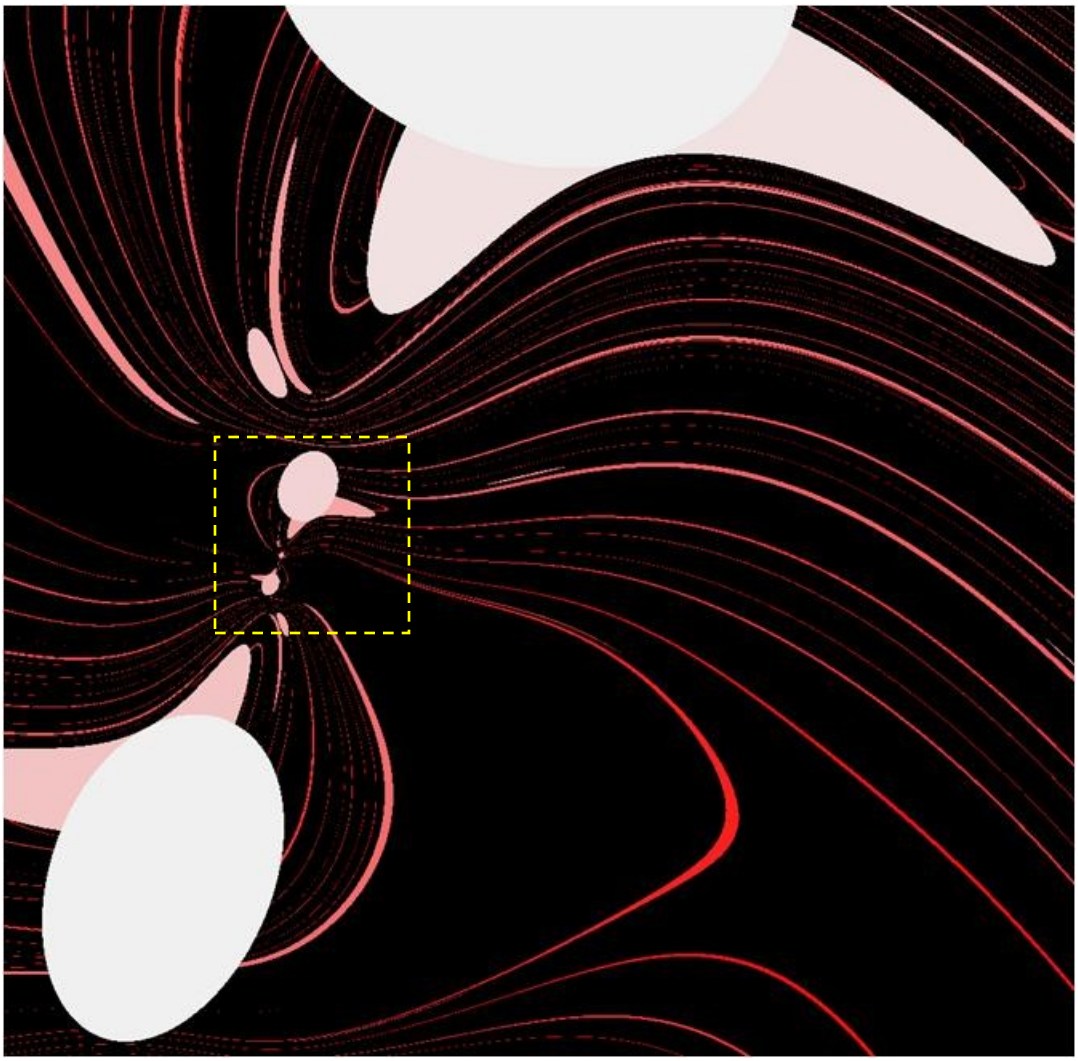


Figure 16. Magnified region of Figure 15 from iteration of $z \leftarrow \left(a_0 + a_7 z^7 + a_{-7} z^{-7}\right)^{-1}$, where complex constants $a_0 = 6$ and $a_7 = 0.0025$ $a_{-7} = 1$. The upper-most small island near 12:30 o'clock in Figure 15 is magnified. Numerically the horizontal span of the image ranges from 0.1 to 0.25 on the real axis. The vertical span of the image ranges from 0.7 to 0.85 on the imaginary axis. Prisoner set is black. Far field escape set is gray. Near field escape set is pink.

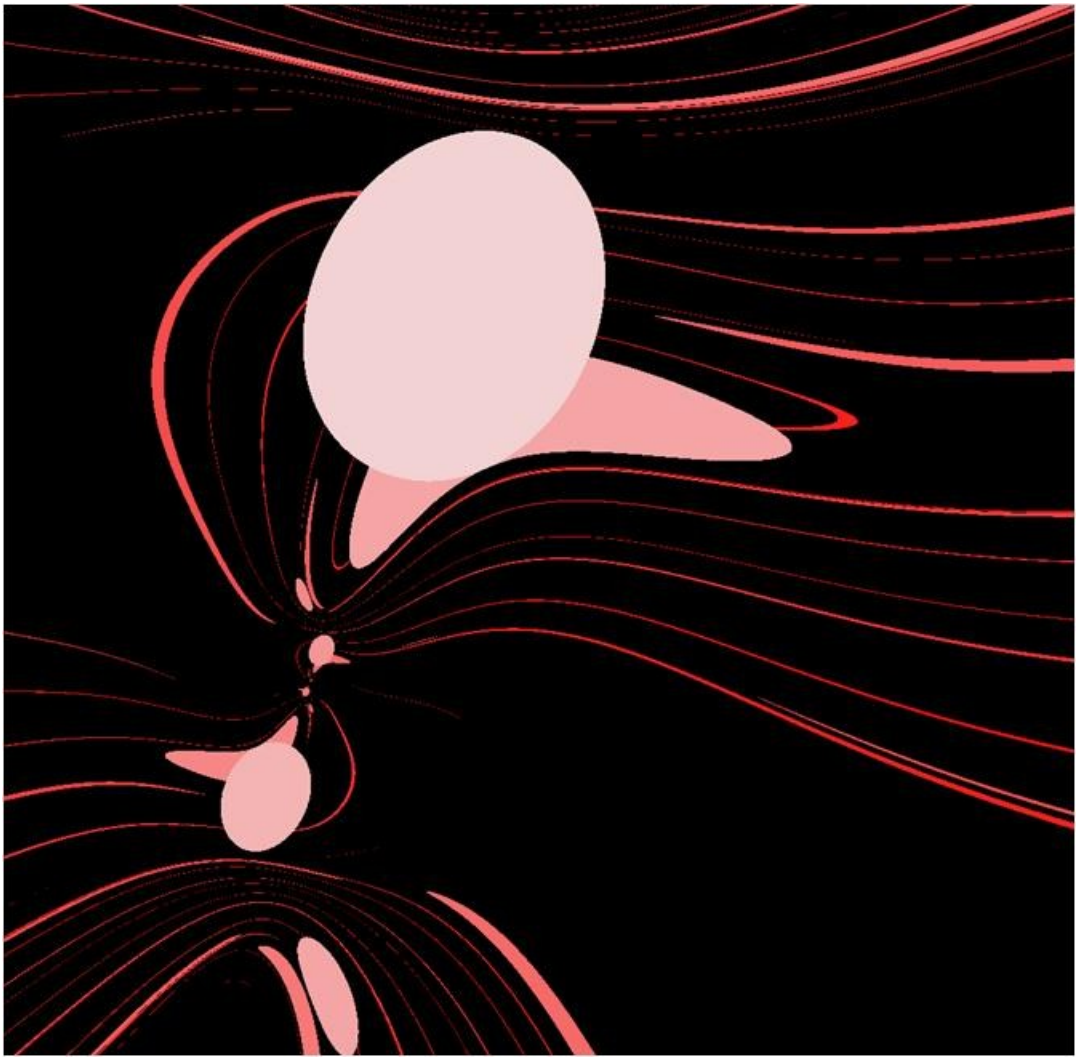
Completely novel forms can arise, as shown in Figure 17(a) for iteration of $z \leftarrow \left(a_2 z^2 + a_1 z + a_0\right)^{-1}$, with $a_0 = a_1 = a_2 = 0 + i$. The range of the graphical map extents from –2 to 2 on the real, horizontal axis and from –2 to 2 on the imaginary, vertical axis. The fractal nature of the pattern is revealed by successive magnifications In Figures 17(b) and 17(c). Dashed yellow boxes indicate approximate regions of further magnification.



(a) $-2 < x < 2$; $-2 < y < 2$



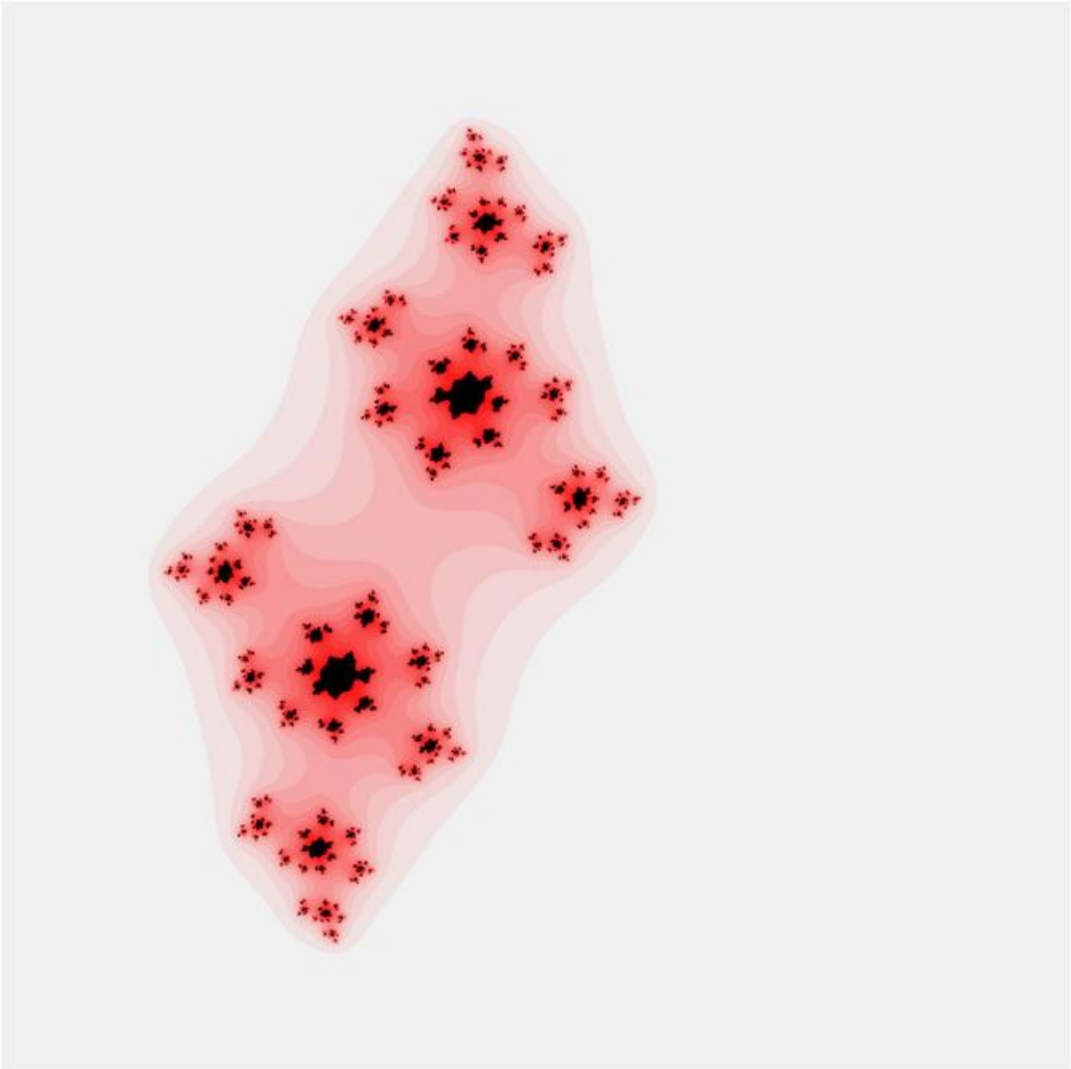
(b) $-1 < x < 0$; $-2 < y < -1$



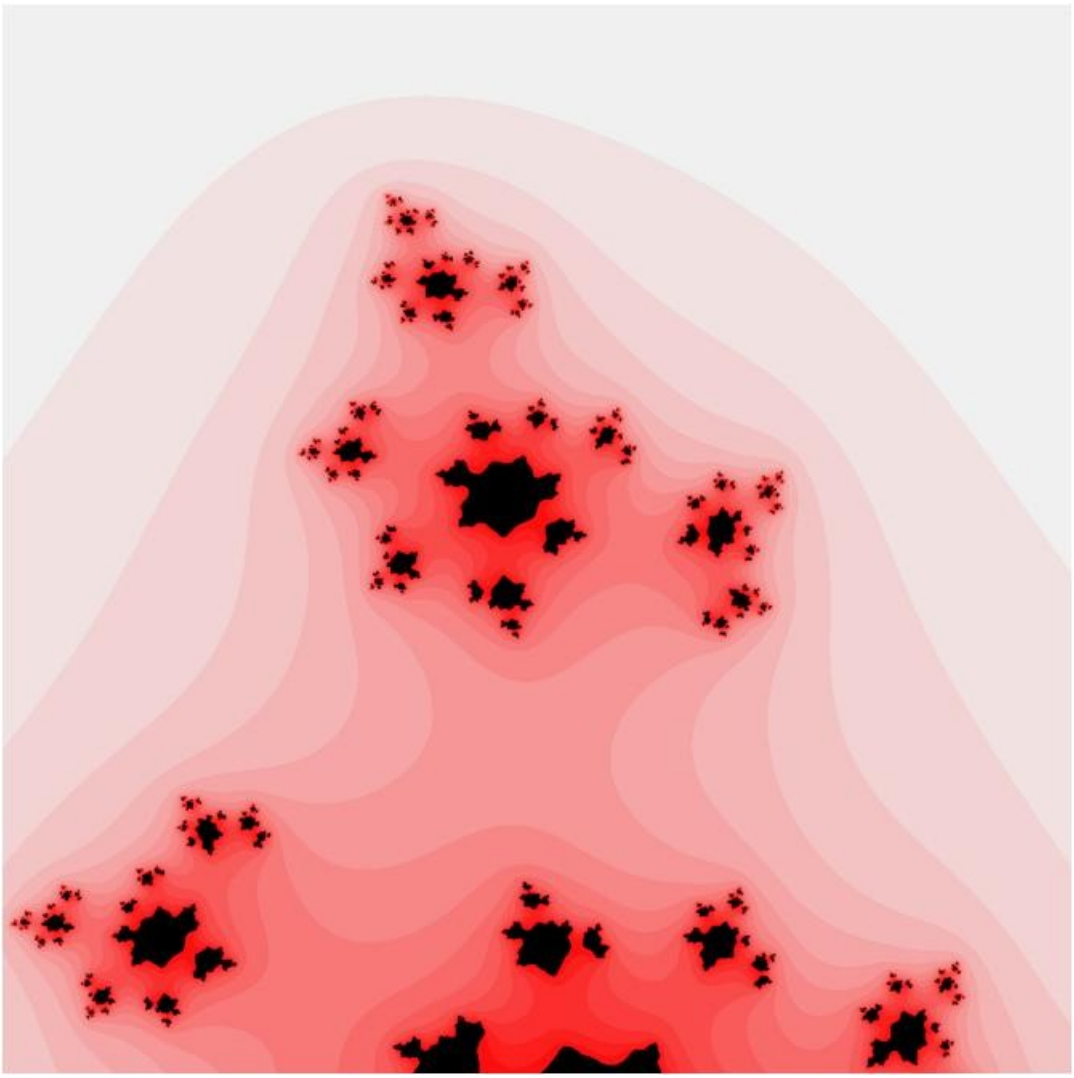
(c) $-0.8 < x < -0.6$; $-1.6 < y < -1.4$

Figure 17. Iteration of $z \leftarrow \left(a_2 z^2 + a_1 z + a_0\right)^{-1}$ for $a_0 = a_1 = a_2 = 0 + i$ at increasingly higher magnification of self-similar regions. Other details similar to Figure 15. Dashed yellow boxes indicate successively magnified regions.

The non-inverted iteration equation, $z \leftarrow a_2 z^2 + a_1 z + a_0$, for $a_0 = a_1 = a_2 = 0 + i$, also produces a fractal pattern, which has an overall dragon shape (Figure 18 (a)). The top head of the dragon is magnified in Figure 18(b).



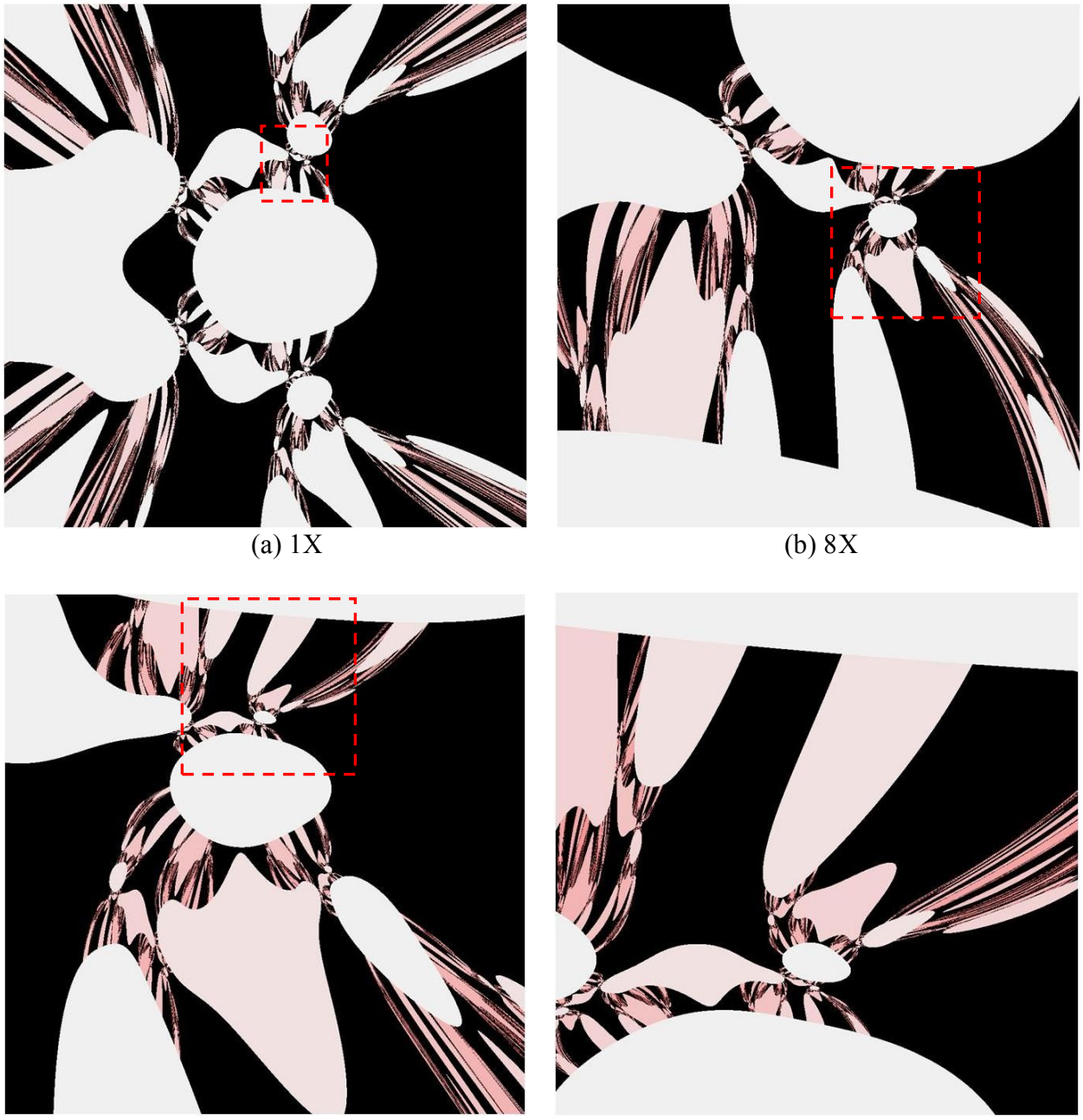
(a) $-2 < x < 2$; $-2 < y < 2$



(b) –0.4 < x< 0; 1.2 < y< 1.6

Figure 18. Iteration of $z \leftarrow a_2 z^2 + a_1 z + a_0$, for $a_0 = a_1 = a_2 = 0 + i$. Details similar to Figure 15.

One more unearthly form is shown in Figure 19 at magnifications ranging up to 80X. The iteration formula is $z_{n+1} = \left[z_n^2 + z_n + 1 + z_n^{-1} + z_n^{-2}\right]^{-1}$. The range in Figure 19(a) is from –2 to 2 in the horizontal and vertical dimensions.



(a) 1X (b) 8X

(c) 27X (d) 80X

Figure 19. Iteration of $z_{n+1} = \left[z_n^2 + z_n + 1 + z_n^{-1} + z_n^{-2}\right]^{-1}$. Colors similar to Figure 15. Dashed red boxes show regions of successive magnification. R = 7, n_{max} = 20.

Discussion

Many thinkers, who are cited in the references, have sought to extend the notion of fractal behavior in the Mandelbrot set to higher-order polynomials. For example, Branner and Hubbard (1988) studied the iteration of cubic polynomials such as $z_{n+1} = z_n^3 - 3a^2 z_n + b$. Frame (1992) studied cubic and quartic polynomial functions of complex numbers as fractal generators. Pickover (1985), Gujar and Vhavsar (1991), Rani and Kumar (2009), and Wang, Jia, and Sun (2009) studied fractals generated by $z_{n+1} = z_n^k + c$ that had circular saw like, planetary, and floral patterns. Zhirriff (1993) studied fractals generated by $z_{n+1} = 1/z_n^k + c$ for negative powers, k, of complex numbers. Wang, Chang, and Gu (2007) studied $z_{n+1} = z_n^k + z_n + c$. Cheng and Tan (2007) studied fractals generated from higher order Mandelbrot-like functions of the form f(z) = z^k + c, for k = any integer, and specifically calculated fractal patterns for k ranging from –8 to +9. Devaney and Look (2005) studied fractals generated by $z_{n+1} = z_n^2 + c/z_n$, and Entwistle (1989) studied a variety of simpler polynomial and trigonometric functions of the complex variable as fractal generators.

The present empirical study extends this work to higher order polynomial functions that include both positive and negative integer powers of a single complex variable. Fascinating structures unlike any exhibited by Julia sets of $z \leftarrow z^2 + c$ can be seen for polynomials including both positive and negative exponents. Additional complexity and richness is easily provided at minimal computational cost by, in turn, raising the resulting polynomial to a positive or negative integer power. This approach further expands the universe of possible fractal forms. A new continent of possibilities awaits exploration by fractal enthusiasts.

Appendix: Pseudocode for key subroutines

Pseudocode is an informal, high-level, easily readable description of what a computer program or algorithm must do that is cast in the structure of a typical programming language, but is intended for human rather than machine reading. The human reader can then translate the pseudocode to machine readable and executable code in a favorite programming language, such as Fortran, C, Visual Basic, Matlab, or Python. Here are pseudocode listings for some critical subroutines used to create the fractals illustrated in this paper.

```
/* return the real, x-component of complex multiplication (x1 + i*y1)*(x2 + i*y2) */

Function cmult_x(x1, y1, x2, y2)

   cmult_x = x1 * x2 - y1 * y2

End Function
```

```
/* return the imaginary, y-component of complex multiplication (x1 + i*y1)*(x2 + i*y2) */

Function cmult_y(x1, y1, x2, y2)

   cmult_y = x1 * y2 + y1 * x2

End Function


/* compute the first n complex powers of z = a + bi, for n > 0 and store the real and imaginary
parts in vectors cPowx() and cPowy() */

Subroutine getPowers(n, a, b)

   cPowx(1) = a
   cPowy(1) = b

   For k = 2 To n
      cPowx(k) = cmult_x(cPowx(k - 1), cPowy(k - 1), a, b)
      cPowy(k) = cmult_y(cPowx(k - 1), cPowy(k - 1), a, b)
   Next k

End Subroutine


/* invert complex number x + i*y */

x = x/(x*x + y*y)
y = -y/(x*x + y*y)
```

References

BODIL BRANNER and JOHN H. HUBBARD, The iteration of cubic polynomials, Acta Mathematica 160, 143-206, 1988

CHENG Jin, TAN Jian-rong, Generalization of 3D Mandelbrot and Julia sets, Journal of Zhejiang University SCIENCE A 8(1), 134-141, 2007

Robert L. Devaney Daniel M. Look, BURIED SIERPINSKI CURVE JULIA SETS, DISCRETE AND CONTINUOUS DYNAMICAL SYSTEMS 13(4), 1035-1046, 2005

IAN D. ENTWISTLE, JULIA SET ART AND FRACTALS IN THE COMPLEX PLANE, Computers & Graphics 13(3), 389-392, 1989

MICHAEL FRAME and JAMES ROBERTSON, A GENERALIZED MANDELBROT SET AND THE ROLE OF CRITICAL POINTS, Computers & Graphics 16(1), 35-40, 1992

UDAY G. GUJAR and VIRENDRA C. BHAVSAR, FRACTALS FROM $z \leftarrow z^{\alpha} + c$ IN THE COMPLEX c-PLANE, Computers & Graphics 15(3), 441-449, 1991

ALAN NORTON, JULIA SETS IN THE QUATERNIONS, Computers and Graphics 13(2), 267-278, 1989

C.A. Pickover, Biomorphs: Computer Displays of Bidogical Forms Generated from Mathematical Feedback Loops, Computer Graphics Forum 5, 313-316, 1986

CLIFFORD A. PICKOVER and ELAHE K.HORASANI, COMPUTER GRAPHICS GENERATED FROM THE ITERATION OF ALGEBRAIC TRANSFORMATIONS IN THE COMPLEX PLANE, Computers and Graphics 9(2), 147-151, 1985

MAMTA RANI and MANISH KUMAR, Circular Saw Mandelbrot Set, RECENT ADVANCES in APPLIED MATHEMATICS December 2009, 131-136, ISBN: 978-960-474-138-0

RAUL ROJAS, A TUTORIAL ON EFFICIENT COMPUTER GRAPHIC REPRESENTATIONS OF THE MANDELBROT SET, Computers & Graphics 15(1), 91-100, 1991

KEN W. SHIRRIFF, AN INVESTIGATION OF FRACTALS GENERATED BY $z \rightarrow 1/z\text{^}n + c$, Computers & Graphics 17(5), 603-607, 1993

JC Sprott and CA Pickover, Automatic generation of general quadratic map basins, Computers & Graphics 19(2), 309-313, 1995

Wang Xing-yuan, Chang Pei-jun, and Gu Ni-ni, Additive perturbed generalized Mandelbrot–Julia sets, Applied Mathematics and Computation 189, 754–765. 2007

Xingyuan Wang, Ruihong Jia, and Yuanyuan Sun, The Generalized Julia Set Perturbed by Composing Additive and Multiplicative Noises, Discrete Dynamics in Nature and Society Volume 2009, Article ID 781976, 18 pages.